In re Application of: Irun R. COHEN et al
Serial No.: 10/542,591
Filed: March 14, 2006
Office Action Mailing Date: March 2, 2011

Examiner: Edward MARTELLO
Group Art Unit: 2628
Attorney Docket: **30070**
Confirmation No.: 6831

REMARKS

Reconsideration of the above-identified application in view of the amendments above and the remarks following is respectfully requested.

Claims 1-33 are in this Application. Claims 13-33 have been withdrawn from consideration. Claims 1-10 and 12 have been rejected under 35 U.S.C. *§* 102. Claim 11 has been rejected under 35 U.S.C. § 103.

Claim 1 is amended.

Rejections Of The Claims

35 U.S.C. § 103 Rejections

The Examiner rejected claims 1-1 12 under 35 U.S.C. §103 (a) as being obvious over Perlin et al in light of Mochizuki et al 684. The rejections of the Examiner are respectfully believed to have been overcome in light of the above amendment and the following arguments.

Perlin indeed teaches two tools, one is an animation engine and the other is a behavior engine.

The present invention teaches a behavior engine that implements behavior as a non-linear series of paths between different behavior states.

Perlin by contrast teaches a behavior engine that "enables authors to create sophisticated rules governing how actors communicate, change and make decisions" (Perlin abstract).

The present claim requires that these two tools run together on the first environment. In the second environment, real time animation is carried out using the animation sequences of the first tool put together with the behavior of the second tool.

In re Application of: Irun R. COHEN et al
Serial No.: 10/542,591
Filed: March 14, 2006
Office Action Mailing Date: March 2, 2011

Examiner: Edward MARTELLO
Group Art Unit: 2628
Attorney Docket: **30070**
Confirmation No.: 6831

As the Examiner concedes, there is nothing in Perlin that relates to any of the features of the second environment.

Indeed the behavior model of Perlin could not run on the second environment. All the behavior model of Perlin is, is a programming tool to allow authors to make the behaviors.

Likewise the animation tool is not a series of animation primitives as per the present invention. Rather it is a tool that allows authors to carry out programming.

Thus Perlin teaches nothing that could be used in a first environment to allow detection of behaviors in a second environment to lead to actual animation to take place.

All it teaches is a programming tool for authors.

Mochizuki is cited as teaching something that could use the tools of Perlin to carry out the features of the second environment of the claim.

However Perlin does not provide the necessary input. It provides no animation primitives and no actually implemented behavior model, so that the stages in Fig. 5 of Mochizuki would have no data to run on.

Mochizuki is concerned with minimizing computational load on the animation side, in their case the client side. That is, Mochizuki ensures that as much computation is carried out on the server side.

Mochizuki does not mention any reactive system abilities. All that Mochizuki does when detecting an event is that he makes a judgment about the time (process 69) and then considers whether to enter process 70 and alter the time line or process 72 and insert a connecting motion.

That is to say, Mochizuki does not teach any creation of animation "on the fly". Rather they dissociate animation from computation, but the animation still has to be pre-specified in some manner. So the the overall animation in Mochizuki is known, all that is changed is the time scale and connecting motion – see seventh and eighth stages.

By contrast, in reactive animation as presently claimed, it is actually impossible to know what the animation might look like, as different objects may initialize different animation sequences, whose combined effects will generate animated objects of higher complexity.

Mochizuki fails to teach a behavior model (a reactive system) which specifies the behavior of the whole system within a single overall flow. A reactive system visualizes the concept of the system by using diagrammatic tools that do not only visually portray behavior but are also machine legible and therefore executable.. Such a reactive system can be implemented, for example by using Unified Modeling Language (UML) which is a language used in the art for specifying large complex systems.

Page 3 lines 15-18 of the present application states: "the present invention therefore preferably enables a reactive system to be connected to an animation tool, for producing true reactive animation." The same page line 21 states "the combination provides a vivid representation built on a rigorous, hard core model of the system under description."

The Examiner points to Fig. 5 processes 61 to 78 as referring to the claimed features of detecting events, selecting animation primitives and putting the primitives together using the behavior model.

Mochizuki processes 61 to 78, as cited by the Examiner, mention detecting events at the client computer and the detection of a state transition of basic behavior. However Mochizuki says nothing about animation primitives and says nothing about a behavior model or about putting animation primitives together using a behavior model.

Rather what it teaches is that a behavior state is entered and then skeletal modifications are applied to a character, say to carry out walking. Walking is carried out by calculation of particular skeletal positions at given times – stages 71 and 74, with interpolation of intermediate positions between the two calculated positions at stage 73 – connection motion.

It is stressed here that Mochizuki does not teach animation primitives in stages 71 and 74 but rather teaches using basic motion data of a full animation to set timings.

There is no animation primitive.

This is not the same as obtaining animation primitives from an animation model and putting them together using a behavior model.

In any event, as argued above, Perlin does not provide the data for Mochizuki to be able to carry any kind of automatic animation.

Figure 5 of Mochizuki does teach that processes 61 to 64 are carried out on the server and processes 65 to 78 are carried out on the client computer.

In fact the combination of Perlin and Mochizuki teaches the following:

Perlin provides an animation tool for use by authors, and behavior processes which the authors can put together with their animations.

Mochizuki teaches that predetermined animations may be calculated at different times with connecting motions in between, following transitions to a new behavior state.

However Perlin would not provide the behavior state to Mochizuki or the animations. Perlin would merely provide tools to an author to make an animation. If the author has made the animation then the additional steps of Mochizuki are redundant. If the author has not made the animation then Mochizuki does not receive the data it needs to work.

Either way the claimed conditions are not met by the combination of Mochizuki and Perlin.

Specifically the combination of Perlin and Mochizuki does not teach:

In a first environment:

> *providing a reactive model of system overall behavior* ; and
>
> *creating animation primitives for animating said model, using a first tool for implementing said animation primitives,*

implementing said reactive model of system overall behavior using a second tool, said second tool being detached from said first tool ;

and

In a runtime environment, said runtime environment being a different environment from said first environment:

detecting events associated with said system;

selecting respectively animation primitives according to said model of overall system behavior and said events; and

combining together said respective animation primitives representing said detected events; thereby to create an overall animation.

since neither citation actually implements a behavior model, nor provides animation primitives which are connected together in accordance with the dictates of events and the behavior model. Rather Perlin merely provides tools for the author to provide animations encoded with behaviors and Mochzuki merely teaches taking pre-set data from the server and calculating timings and inbetween motions to be applied to the data at the server.

But Perlin does not provide the predetermined data (Mochizuki stages 61 and 62) so the two could not work together, and in any event Mochizuki fails to provide the hint that the predetermined data should be animation primitives and an implemented behavior model. Perlin does not hint at these either since implementation and animation are left to the authors.

In view of the above amendments and remarks it is respectfully submitted that claim 1 is novel and inventive over the cited prior art. Claims 2 – 12 depend form allowable claim 1.

Hence claims 1-12 are now believed to be in condition for allowance. A prompt notice of allowance is respectfully and earnestly solicited.

Respectfully submitted,

/Jason H. Rosenblum/

Jason H. Rosenblum
Registration No. 56,437
Telephone: 718.246.8482

Date: July 5, 2011

Enclosures:

- Petition for Extension (One Month)